Amendments to the Claims

This listing of claims will replace all prior versions, and listings, of claims in the application. Please cancel claims 2, 4, 19, 21, 23 and 25 and amend claims 1, 3, 5, 6, 18, 20, 22, 24 and 26.

Listing of Claims:

1. (currently amended) A method of managing memory in a multi-threaded processing environment including local thread stacks and local thread heaps, and a global heap, said method comprising:

creating an object in a thread heap; ~~and~~

for a given thread, monitoring each object in its heap, to determine whether the object is accessible by any thread other than the given thread.~~;~~

assigning a status to the given object, the status designating the object as a local object; and

changing the status of the object to global when the monitoring step determines that the object is accessible from either of a global root or other global object.

2. (cancelled)

3. (currently amended) The method as claimed in claim ~~2~~ 1 further comprising deleting from the thread heap one or more local objects when it is determined that they are not accessible from a local root.

4. (cancelled)

5. (currently amended) The method as claimed in claim ~~2~~ 1, further comprising changing the status of an object in the thread heap to global if the object is assigned to a static variable or if the object is assigned to a field in a global object.

6. (currently amended) The method as claimed in claim ~~2~~ 1, further comprising intercepting assignment operations to an object in the thread heap to determine whether the object status should be changed.

7. (previously presented) The method as claimed in claim 6 wherein the multithreaded processing environment is a virtual machine.

8. (previously presented) The method as claimed in claim 7 wherein the virtual machine comprises an interpreter comprising a write operation code modified to perform a checking of assignment of the object.

9. (previously presented) The method as claimed in claim 8 wherein the virtual machine comprises a just in time compiler wherein native compiled write operation code includes native code to perform the checking of assignment of the object.

10. (previously presented) The method as claimed in claim 9 further comprising using spare capacity in an object header for the status.

11. (previously presented) The method as claimed in claim 10 further comprising using multiples of 2 or more bytes in a thread heap to store the objects whereby there is at least one spare bit in the object length variable and using the at least one spare bit as the status.

12. (previously presented) The method as claimed in claim 11 further comprising moving objects whose status is global from the thread heap to the global heap.

13. (previously presented) The method as claimed in claim 12 further comprising compacting the reachable local objects in a thread heap.

14. (previously presented) The method as claimed in claim 1 wherein certain objects are associated with a global status on creation thereof.

15. (previously presented) The method as claimed in claim 14 where said certain objects include class objects.

16. (previously presented) The method as claimed in claim 14 further comprising the step of analysing whether an object is likely to be made global and associating such an object with a global status on creation.

17. (previously presented) The method as claimed in claim 16 further comprising allocating objects assigned as global on creation to the global heap.

18. (currently amended) A system for managing memory in a multi-threaded processing environment comprising:

respective local thread stacks and heaps;

a global heap;

means for creating an object in a thread heap; ~~and~~

means for monitoring each object in its heap, to determine whether the object is accessible by any thread other than the given thread

means for associating a local status with the object and means for changing the status of the object to global under certain conditions; and

means for changing the status of an object in the thread heap to global if the object is assigned to a static variable or if the object is assigned to a field in a global object.

19. (cancelled)

20. (amended) The system as claimed in claim ~~19~~ 18 further comprising means for deleting from the thread heap one or more local objects when they are not accessible from a local root.

21. (cancelled)

22. (currently amended) A computer program product stored on a computer readable storage medium for, when executed on a computer, performing a method of managing memory in a multi-threaded processing environment including local thread stacks and local thread heaps, and a global heap, said method comprising:

creating an object in a thread heap; ~~and~~

for a given thread, monitoring each object in its heap, to determine whether the object is accessible by any thread other than the given thread;

assigning a status with the object that designates the object as a local object;

changing the status of the object to global when the monitoring step determines that the object is accessible from either of a global root or other global object.

23. (cancelled)

24. (currently amended) The computer program product as claimed in claim ~~23~~ 22 further comprising deleting from the thread heap one or more local objects when it is determined that they are not reachable from a local root.

25. (cancelled)

26. (currently amended) The computer program product as claimed in claim ~~23~~ 22 wherein the status of an object in the thread heap is changed to global if the object is assigned to a static variable or if the object is assigned to a field in a global object.